Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Model-Based Systems Engineering

System Modeling, Testing and Simulation with UML/SysML and Modelica

Parham Vasaiely
December 2009

Table of Contents

1. Introduction and Motivation (Systems-Engineering)

Problem: Increasing complexity of technical systems

- -Increased degree of complexity -> need for mastering complexity by e.g. automation
- -Increased number of involved technologies
- -SW and HW development is strongly coupled
- -Component suppliers are involved in system design
- -Need for system validation long before first physical prototypes are built

Motivation:

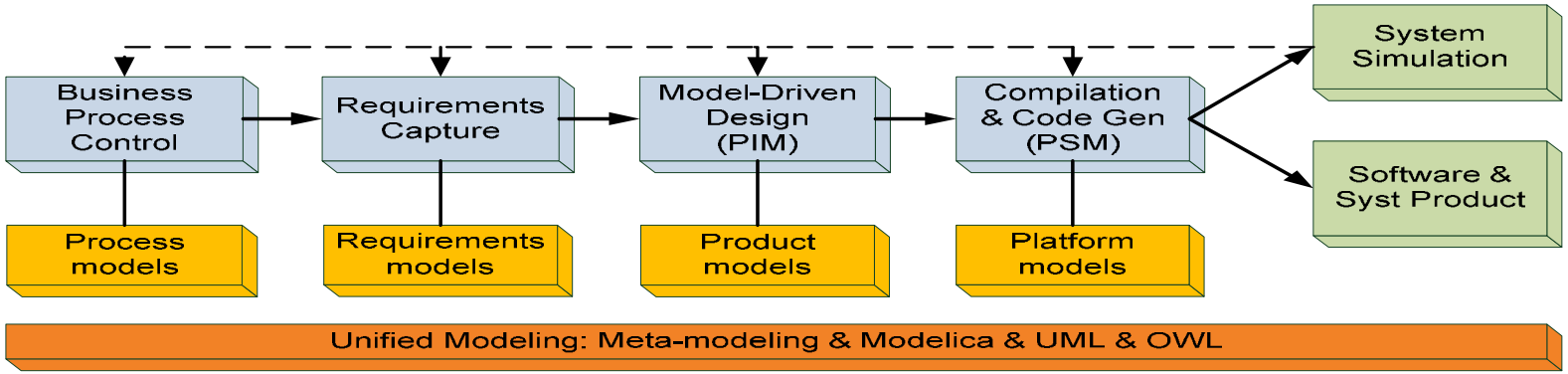
- -Process efficiency and development time and cost reduction in systems development
- -Time to market reduction
- -Ensure quality and maturity of system definition in early development stages

Solutions / Enablers:

- -Model- Based Systems Engineering can help to improve development process efficiency
- -System validation and tests by simulation in early development stages

1.1 OPENPROD – Introduction and Goals

Open whole-product model-driven systems development modelling and simulation

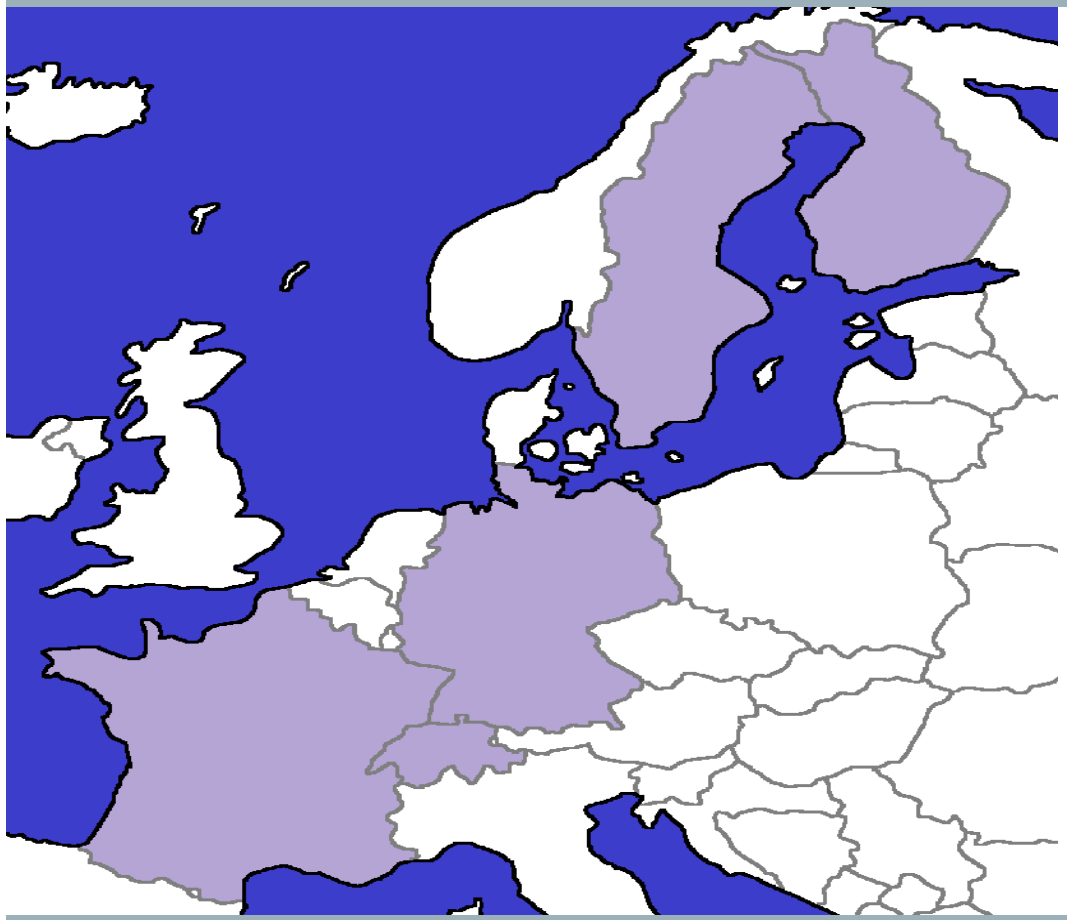


Goals:

-Holistic whole-product model-driven rapid development and design environment
-Including support for product business processes.
-Open source tools and components for open reusable solutions.
-Standardized model representation of products.

Start: June 2009 End: May 2012

1.1 OPENPROD – Involved Partners



EADS
NOKIA
MathCore
eDF
SIEMENS
APPEDGE
Consulting & Engineering
PSA PEUGEOT CITROËN
Fraunhofer FIRST
plexim
Rexroth Bosch Group
PÖYRY

1.3 System Simulation With Modelica

Published by the Modelica Association (www.modelica.org)

Declarative, Object-Oriented and Synchronous Language

- -Equations and mathematical functions allow acausal modeling, high level specification, increased correctness
- -Equation-based; continuous and discrete equations
- -Acausal modeling supports better reuse of modeling and design knowledge than traditional classes

Multi-Domain Modeling

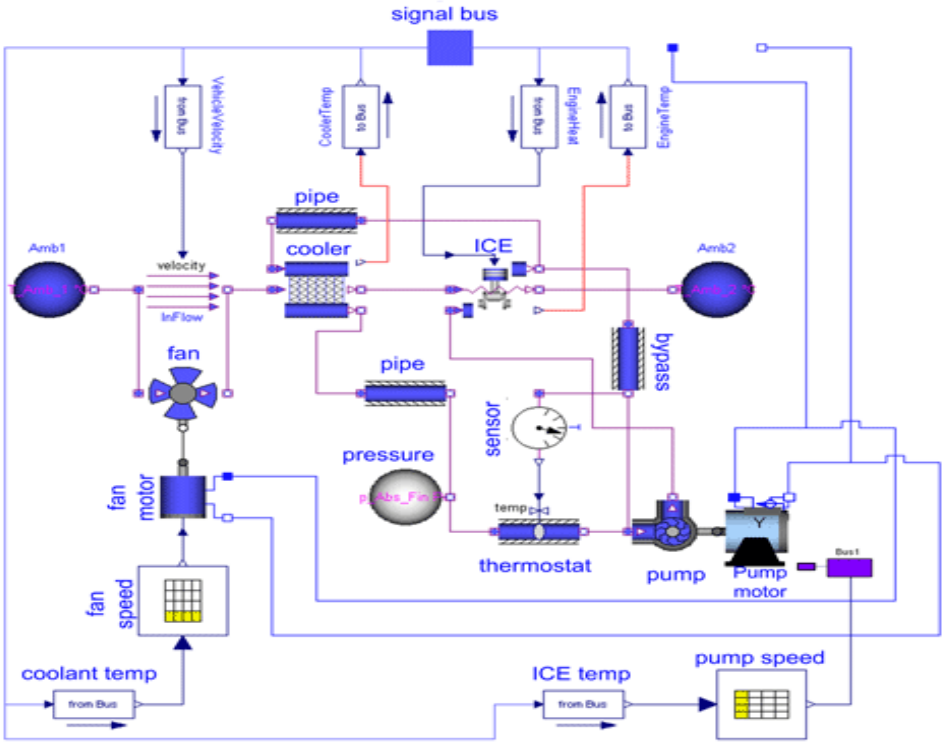
- -Combines electrical, mechanical, thermodynamic, hydraulic, Biomechanik , control, event, real-time, etc...

Efficient, Non-Proprietary

- -Efficiency comparable to C; advanced equation compilation, e.g. 300 000 equations, ~150 000 lines on standard PC

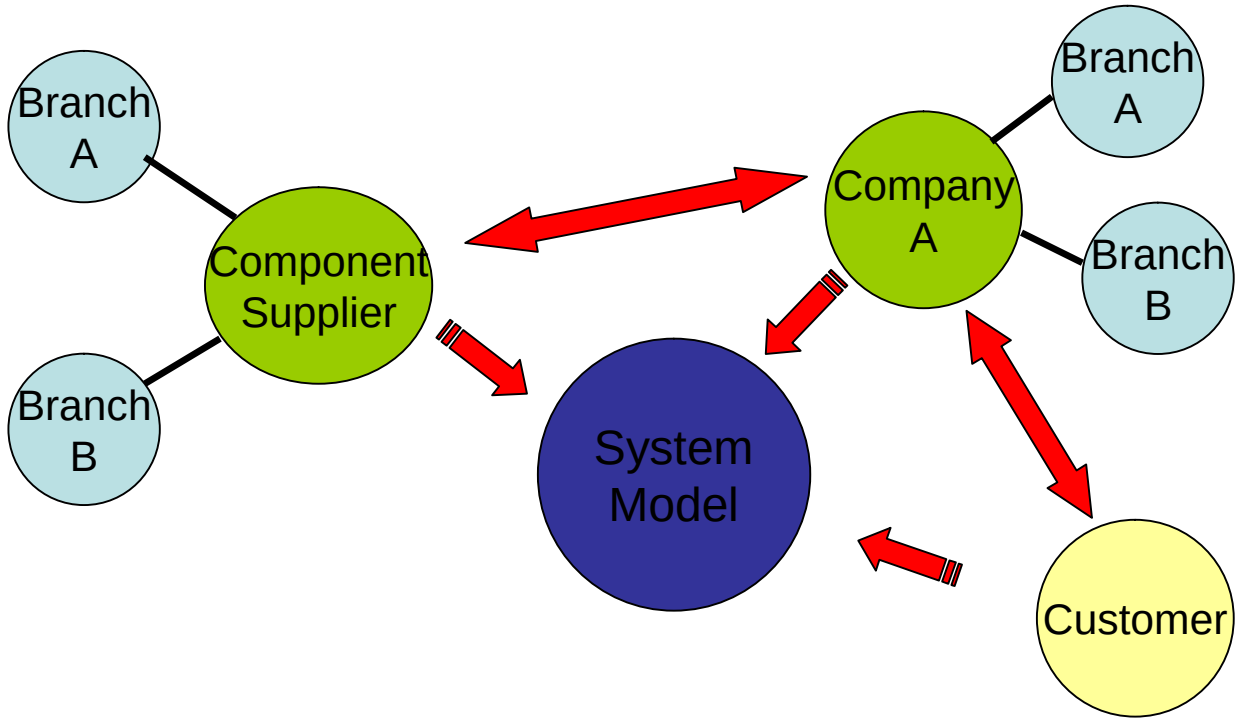
1.2 Need for Standardized Model Representations

Challenge **à** Many domain-specific graphical notations



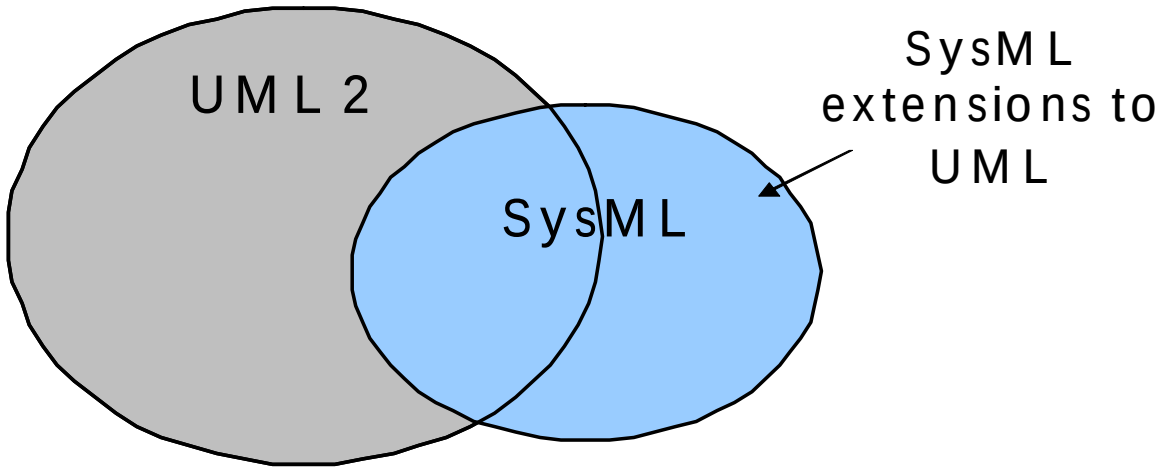
1.2 Need for Standardized Model Representations

Solution **à** Standardized notations for systems modeling



1.2 OMG SysML

OMG Systems Modeling Language

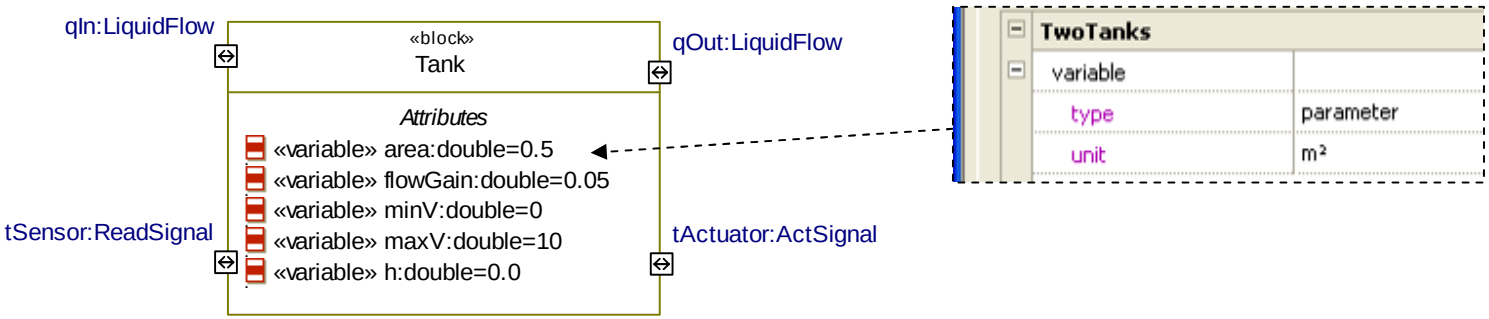


Graphical modeling language to…

Specify… Analyze… Design… Check…

Systems and single System components

1.2 OMG SysML to Modelica Code Transformation

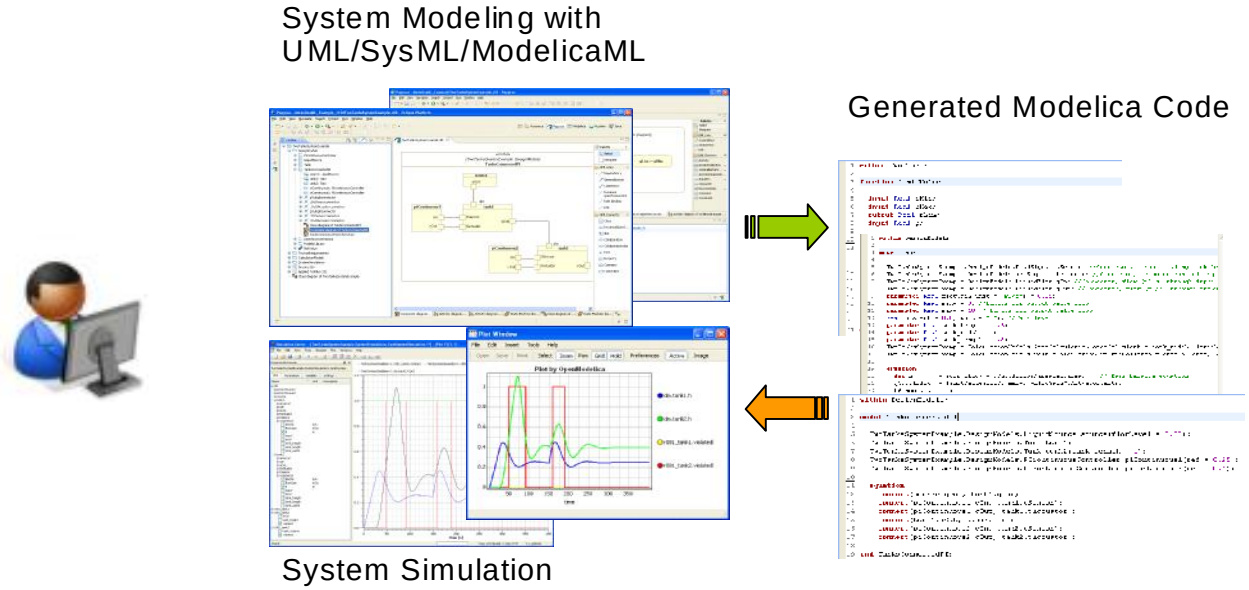


<<block>>Tank **à** Tank.mo (Modelica)

```
within TwoTanks;
block Tank
    ReadSignal tSensor;
    ActSignal tActuator;
    LiquidFlow qIn;
    LiquidFlow qOut;
    parameter Real area (unit = "m2") = 0.5;
    parameter Real flowGain (unit = "m2/s") = 0.05;
    Real h (start = 0.0, unit = "m");
end Tank;
```

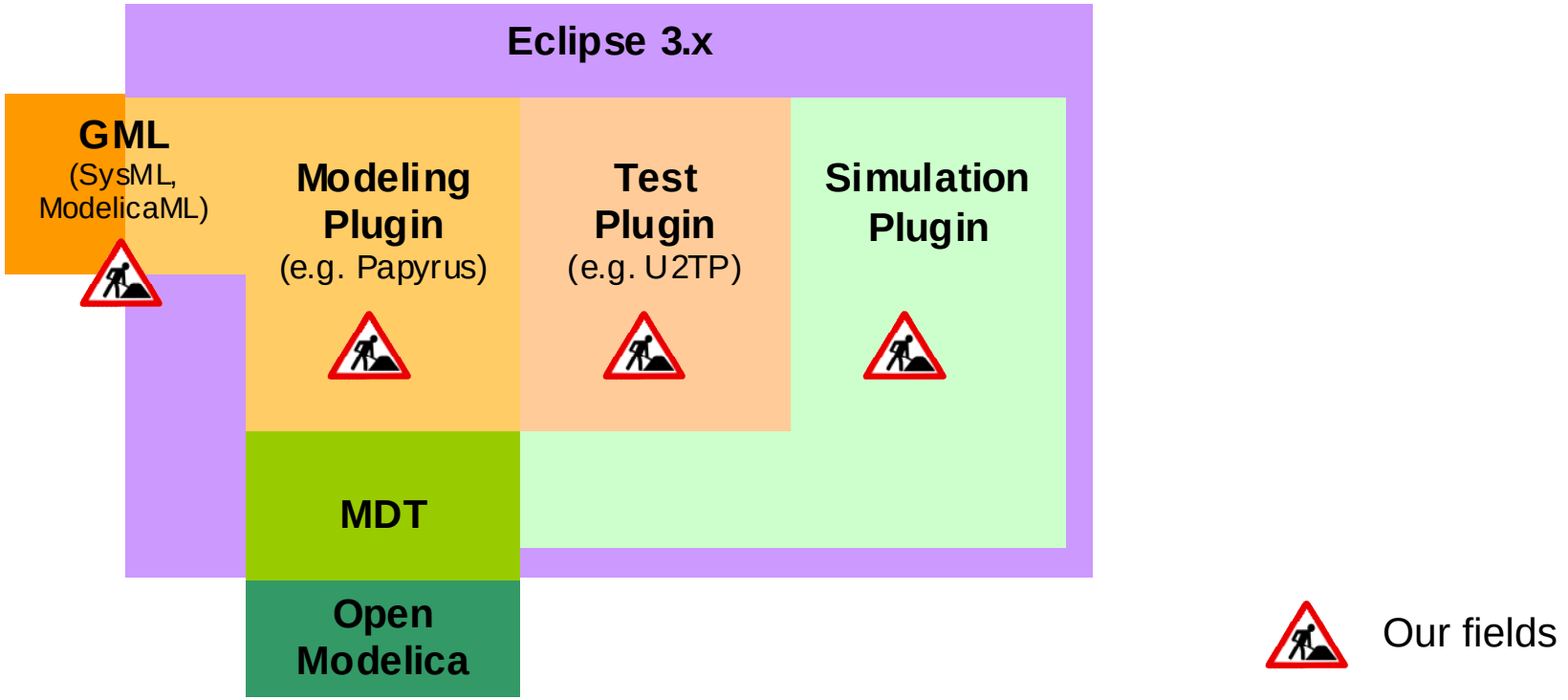
1.5 Integration of UML/SysML and Modelica

Putting together UML/SysML and Modelica gives a powerful combination for modelling and simulation of complex systems at any stage of system development.



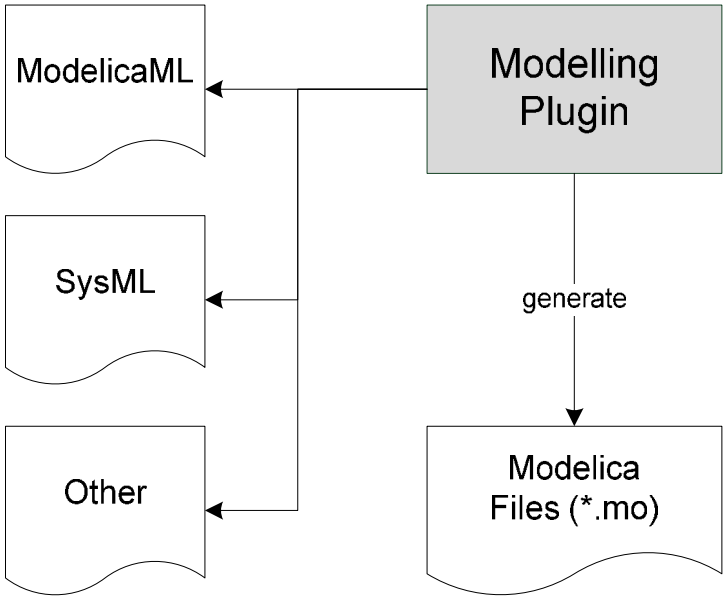
2. Project Fields

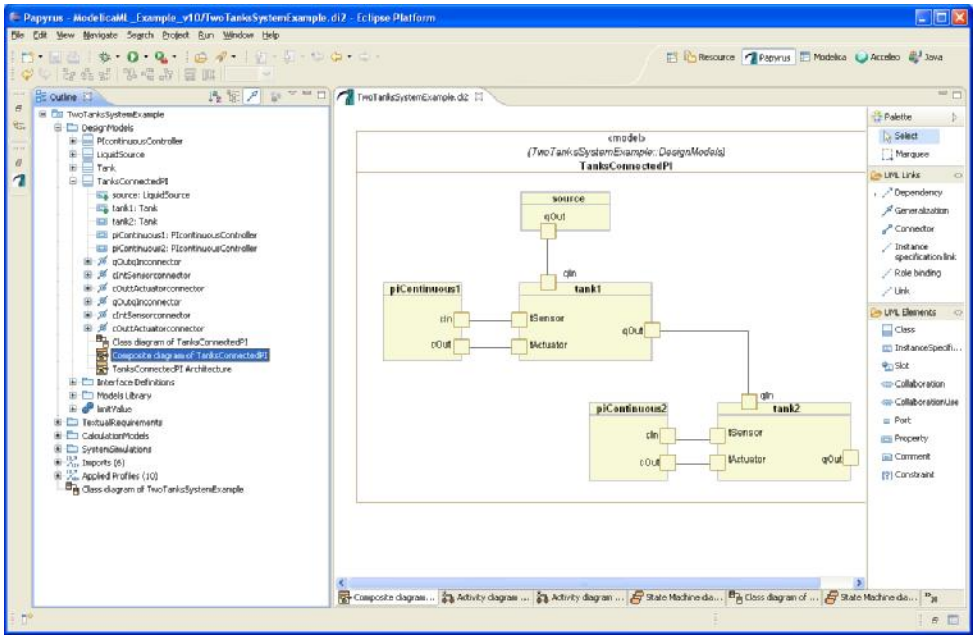
Developing the following environments as Eclipse plug-ins



2.1 The Modeling Environment

System Structure, Behavior, Requirements





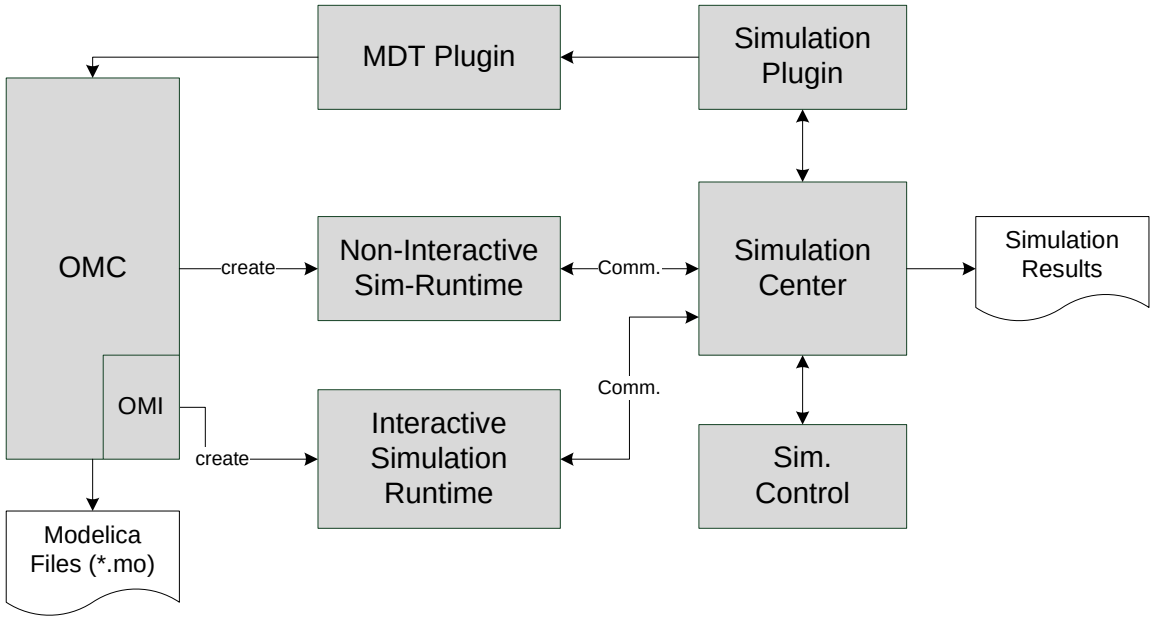
2.2 The Simulation Environment

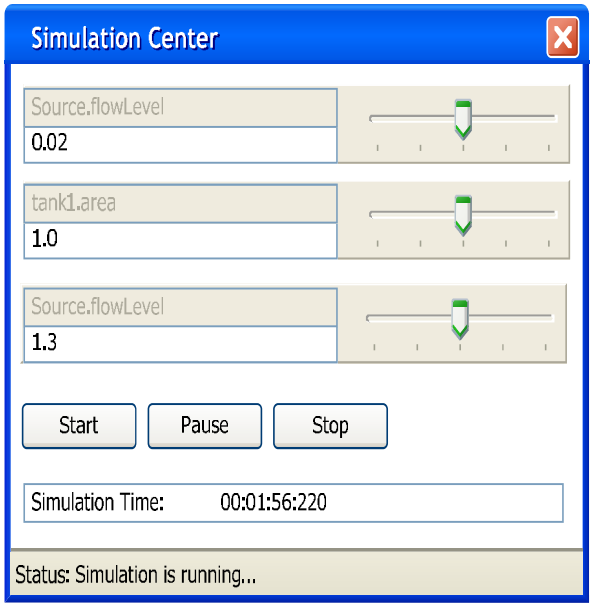
Why is Simulation important?

-First virtual Prototype
-Supports Communication between Developer and Customer
-Basic unit for dynamic system tests
-Interactive system tests

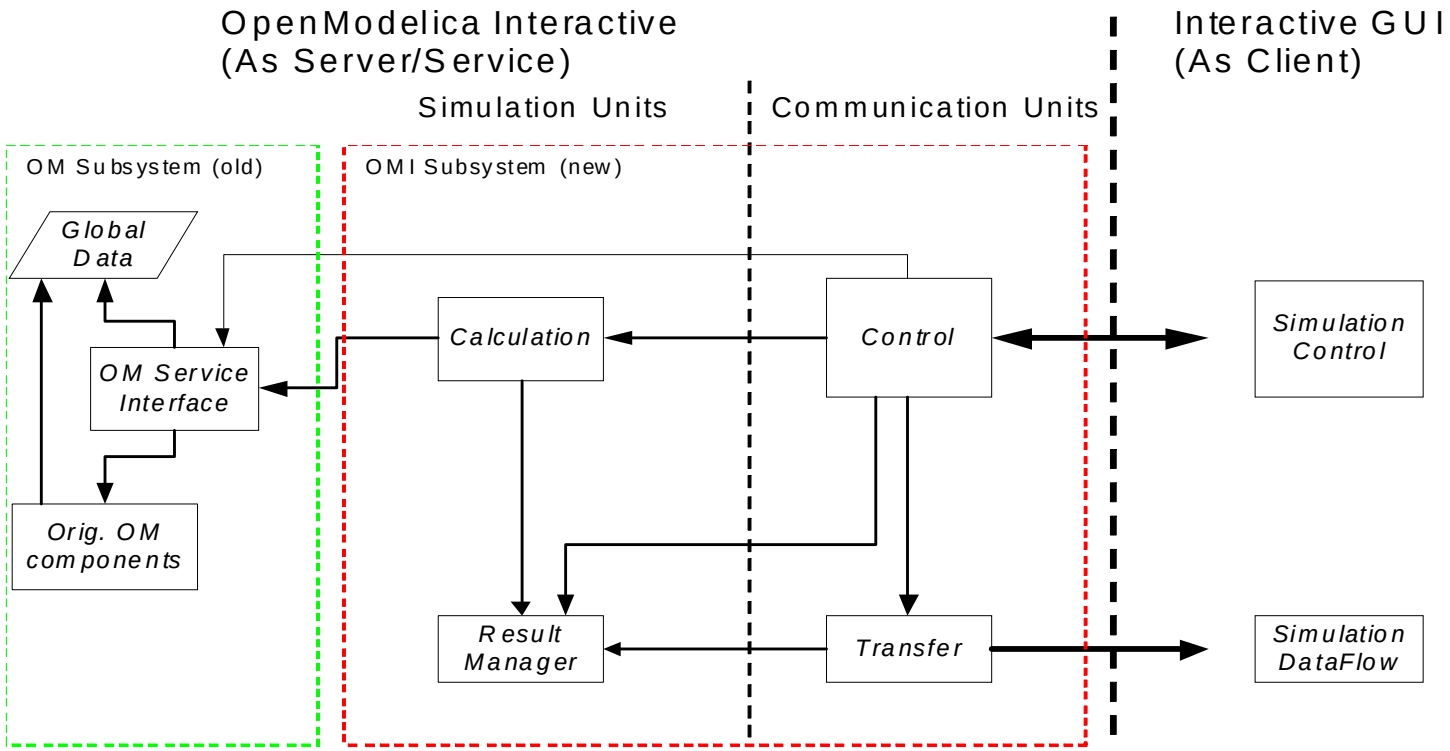
2.2 Simulation Environment Example

Non-/Interactive Simulation

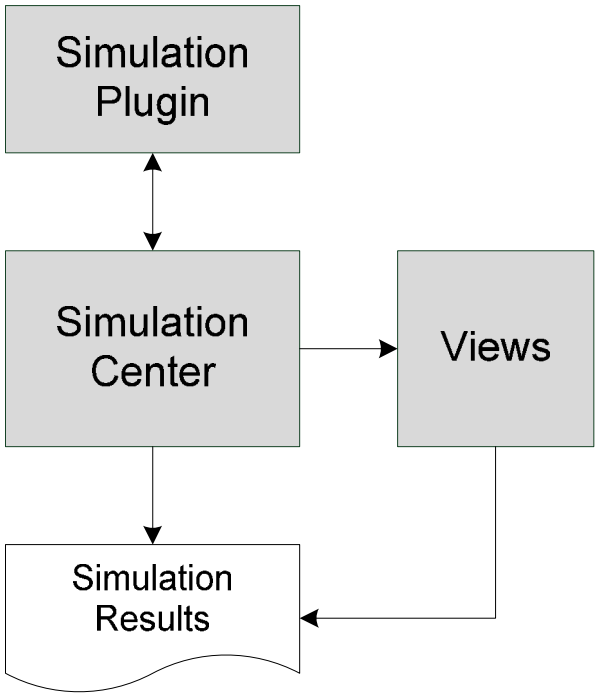


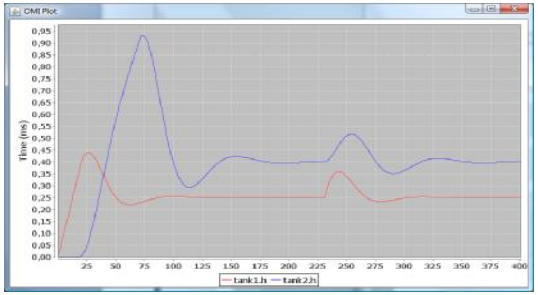


2.2 Interactive Simulation Runtime Example Implementation

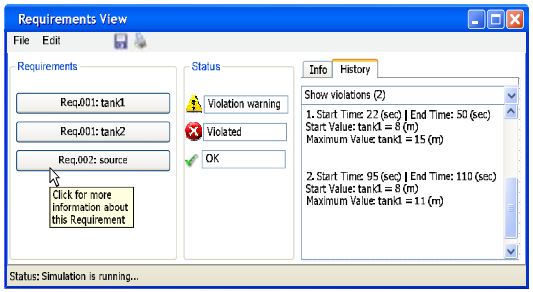


2.2 Simulation Visualization Exmample

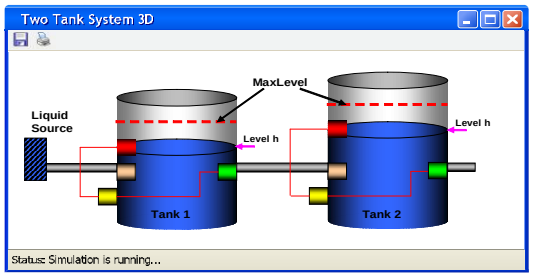




Plot View



Requirements Evaluation View



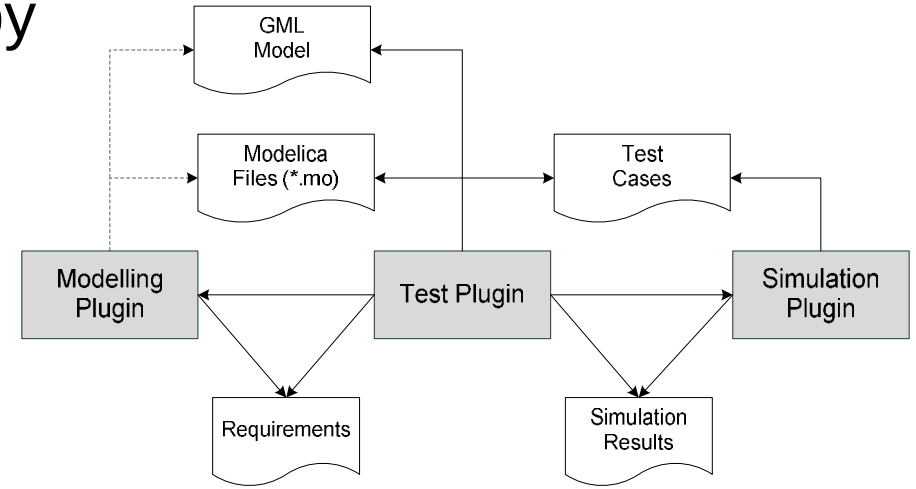
Domain-Specific Visualization View

2.2 Test Environment - Needs

As an addition to the OPENPROD

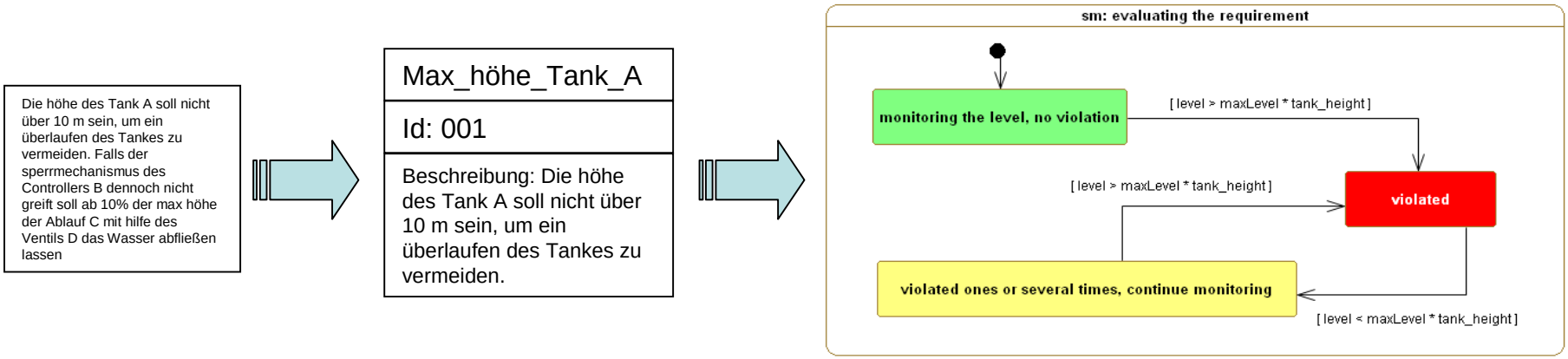
Why is Testing important?

-Searching for a failure caused by fails
-Imporve Quality
-Verifikation and Validation



2.2 Model Based System Dynamics Testing

Model Based: Formal modeling of informal text based requirements

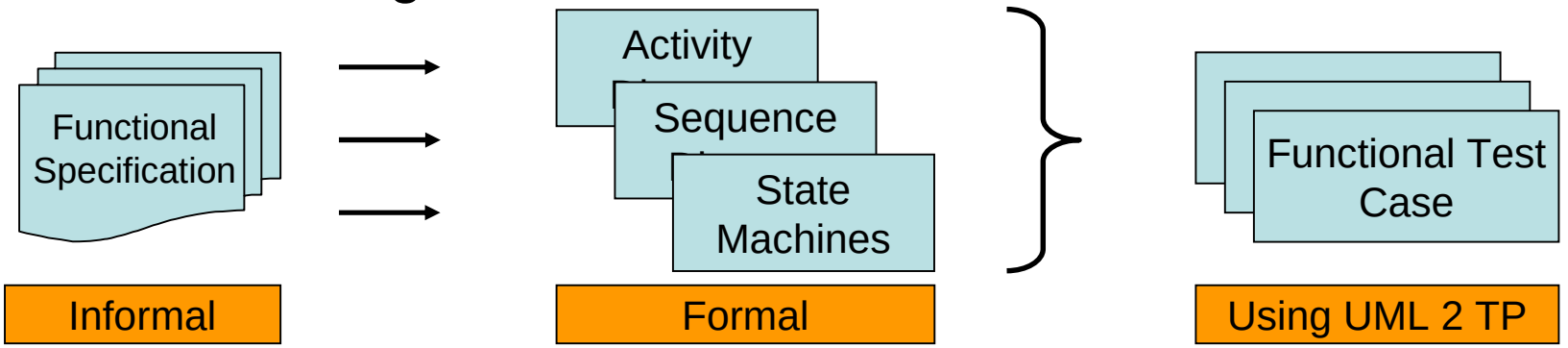


- Activity Diagram, Sequence Diagram, State Machines…
- Visuelle Kontrakte

Dynamic: Using simulation

2.2 Test Automation

1. Test Case generation



2. Test Coverage

3. Test Execution

References

- Sanford Friedenthal, Alan Moore and Rick Steiner, Practical Guide to SysML, 2008
- Fritzson Peter. Principles of Object-Oriented Modeling and Simulation with Modelica 2.1, 2004
- Modelica Association. Modelica: A Unified Object-Oriented Language for Physical Systems Modeling: Language Specification Version 3.0, Sept 2007. www.modelica.org
- OMG. OMG Unified Modeling Language TM (OMG UML). Version 2.2, February 2009.
- OMG. OMG Systems Modeling Language (OMG SysML™), Version 1.1, November 2008.
- OMG. OMG UML 2 Testing Profile (U2TP), Version 1.0, July 2007
- Acceleo, Eclipse Plug-In. www.acceleo.org/pages/home/en
- Papyrus UML, www.papyrusuml.org
- Schamai.W, ModelicaML - UML Profile for Modelica, Technical Report http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-20553
- OPENPROD ITEA2 Project Spec 2009 (Non public version)
- Robert V. Binder. Testing Object-Oriented Systems, Addison-Wesley, 2000.

Thank you for your attention!

Parham Vasaiely
Parham.Vasaiely@informatik.haw-hamburg.de